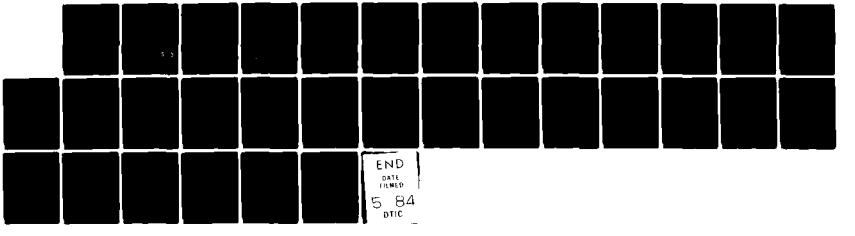
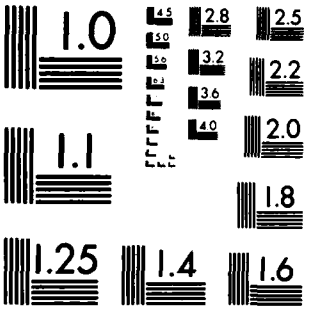
AD-A140 115 ON THE COMPLEXITY OF DECENTRALIZED DECISION MAKING AND 1/1
DETECTION PROBLEMS..(U) MASSACHUSETTS INST OF TECH
CAMBRIDGE LAB FOR INFORMATION AND D..
UNCLASSIFIED J N TSITSIKLIS ET AL. 16 MAR 84 LIDS-P-1366 F/G 5/10 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

LIDS-P- 1366

MARCH 16, 1984

15

ON THE COMPLEXITY OF DECENTRALIZED DECISION MAKING AND DETECTION PROBLEMS†

by

John N. Tsitsiklis*

Michael Athans**

Abstract

The authors ~~We~~ study the computational complexity of the discrete versions of some simple but basic decentralized decision problems. These problems are variations of the classical "team decision problem" and include the problem of decentralized detection, whereby a central processor is to select one of two hypotheses, based on 1-bit messages from two non-communicating sensors. Our results point to the inherent difficulty of decentralized decision making and suggest that optimality may be an elusive goal.

†Research was supported by the Office of Naval Research under contract ONR/N00014-77-C-0532 (NR 041-519).

*Information Systems Laboratory, Department of Electrical Engineering, Stanford University, Stanford, CA 94305.

**Laboratory for Information and Decision Systems, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139.

This paper has been submitted to the 1984 IEEE Decision and Control Conference and to the IEEE Trans. on Auto. Control.

I. INTRODUCTION

The field of decentralized (distributed) decision making has been an active area of research for more than two decades [4, 11, 12]. In the meantime, it has been realized that decentralized decision problems are qualitatively different from the corresponding decision problems with centralized information. The classical "counterexample" of Witsenhausen [5,17] in decentralized stochastic control best illustrates this point. It is safe to conjecture that the prohibitive factor in decentralized problems is not so much the inadequacy of the mathematical tools presently been used, but rather the inherent complexity (in the broad sense of the term) of the problems that are usually been formulated. However, this ever-present complexity is still to be pinned down in a precise mathematical way. The present paper, which follows the line of research of [10], should be viewed as a contribution in this direction. We focus on finite versions of some simple but fairly typical decentralized decision making problems and characterize their complexity by using the tools of the theory of computational complexity [2,9]. Keeping with the tradition of this theory, we consider "easy" those problems that can be solved by a polynomial algorithm, whereas we consider NP-complete (or worse) problems to be "hard". In our opinion, such an approach is a) more satisfying intellectually and b) given the present state of the theory of decentralized decision making, it will allow us to sytematically identify hard problems and redirect research efforts to new, easier or appropriately formulated problems.

Overview.

The main issue of interest in decentralized systems may be loosely phrased as "who should communicate to whom, what, when, etc.". From a purely logical point of view, however, there is a question which precedes the above: "are there any communications necessary"? Section 2 addresses the difficulty of the problem of deciding whether any communications are necessary, for a given decentralized system. We use a formulation of this problem introduced in [10]. We impose some additional structure on this problem and we are able to determine the boundary between easy and hard cases. In Section 3 we formulate the discrete version of the decentralized detection problem and prove that it is, in general, a hard one. In Section 4 we present and discuss a few problems related to the problem of Section 2, including the team decision problem. Section 5 contains our conclusions. All proofs may be found in the Appendix.

II. A PROBLEM OF SILENT COORDINATION

Let $\{1, ..., M\}$ be a set of processors (decision makers). Each processor i obtains an observation $y_i \in Y_i$, where Y_i is a finite set. Then, processor i chooses a decision u_i belonging to a finite set U_i of possible decisions, according to a rule $u_i = \gamma_i(y_i)$, where γ_i is some function from Y_i into U_i. The M-tuple $(y_1,...,y_M)$ is the total information available and may be viewed as the "state of the environment". For each state of the environment, we assume that only certain M-tuples $(u_1,...,u_M)$ of decisions accomplish a given, externally specified goal. More precisely, for each $(y_1,....,y_M) \in Y_1 \times .... \times Y_M$, we are given a set $S(y_1,....,y_M) \subset U_1 \times \cdots \times U_M$ of *satisficing* (using the term introduced by H. Simon [13]) decisions. So, S may be viewed as a function from $Y_1 \times \cdots \times Y_M$ into $2^{U_1 \times \cdots \times U_M}$. We then ask whether there exist decision rules $\gamma_i : Y_i \to U_i$ (involving no communications) which always lead to satisficing decisions. This problem was first introduced (for the case $M = 2$) in [10] and may be formalized as follows:

Problem DS: Given finite sets $Y_1, \ldots, Y_M$, $U_1, \ldots, U_M$ and a function $S : Y_1 \times \cdots \times Y_M \to 2^{U_1 \times \cdots \times U_M}$, are there functions $\gamma_i : Y_i \to U_i$, $i=1,...,M$, such that

$$(\gamma_1(y_1),...,\gamma_M(y_M)) \in S(y_1,...,y_M), \quad \forall (y_1, \ldots, y_M) \in Y_1 \times \cdots \times Y_M \quad ? \tag{2.1}$$

We are assuming above that the function S is easily computable; for example, it may be given in the form of table. Also, note that the centralized counterpart of *DS* would be to allow the decision u_i of each processor depend on the

entire set of observations, in which case the problem would become trivial. Since *DS* has a trivial centralized counterpart, any difficulty inherent in *DS* is only caused by the fact that information is decentralized. In this sense, *DS* captures the essence of coordinated decentralized decision making (silent coordination).

It was shown in [10] that:

a) The problem *DS* with two processors ($M = 2$) and restricted to instances for which the cardinality of the decision sets is 2 ($|U_i| = 2,\ i = 1,2$)* may be solved in polynomial time.

b) The problem *DS* with two processors ($M = 2$) is NP-complete even if we restrict to instances for which $|U_1| = 2,\ |U_2| = 3$.

An extension of the above results is the following:

Proposition 2.1: The problem *DS* with three or more processors ($M \geq 3$) is NP-complete, even if we restrict to instances for which $|U_i| = 2, \forall i$.

We may therefore conclude that the problem *DS* is, in general, a hard combinatorial problem, except for the special case in which there are only two processors and each one has to make a binary decision. It should be noted that the difficulty is not caused by an attempt to optimize with respect to a cost function, because no cost function has been introduced. In game theoretic language, we are faced with a "game of kind", rather than a "game of degree".

We now turn to special cases of *DS*, by introducing some more structure (reflecting the nature of typical practical problems), with the aim of determining

*For any finite set A, we let $|A|$ denote its cardinality.

the dividing line between easy and hard special cases. Moreover, we restrict to the case of only two processors ($M = 2$). (Certainly, problems with $M > 2$ cannot be easier.)

In the formulation of DS that we introduced earlier, all pairs $(y_1,y_2) \in Y_1 \times Y_2$ are likely to occur. So, the information of different processors is completely unrelated; their coupling is caused only by the structure of the satisficing sets $S(y_1,y_2)$. In most practical situations, however, information is not completely unstructured: when processor 1 observes y_1, he is often able to make certain inferences about the value of the observation y_2 of the other processor and exclude certain values. We now formalize these ideas:

Definition: An **Information Structure** I is a subset of $Y_1 \times Y_2$. We say that an information structure I has **degree** (D_1,D_2), (D_1,D_2 are positive integers) if:

(i) For each $y_1 \in Y_1$ there exist at most D_1 distinct elements of Y_2 such that $(y_1,y_2) \in I$.

(ii) For each $y_2 \in Y_2$ there exist at most D_2 distinct elements of Y_1 such that $(y_1,y_2) \in I$.

(iii) D_1, D_2 are the smallest integers satisfying (i), (ii).

An information structure I is called **classical** if $D_1 = D_2 = 1$; **nested** if $D_1 = 1$ or $D_2 = 1$.

We now interpret this definition: The information structure I is the set of pairs (y_1,y_2) of observations that may occur together. If I has degree (D_1,D_2),

processor 1 may use his observation y_1 to decide which elements of Y_2 may have been observed by processor 2. In particular, he may exclude all elements except for (at most) D_1 of them. The situation faced by processor 2 is symmetrical.

If $D_1 = 1$ and processor 1 observes y_1, there is only one possible value for y_2. So, processor 1 knows the observation of processor 2. (The converse is true when $D_2 = 1$). This is called a nested information structure because the information of one processor contains the information of the other. When $D_1 = D_2 = 1$, each processor knows the observation of the other; so, their information is essentially shared.

By restricting our attention only to pairs $(Y_1, Y_2) \in I$, we obtain the following version of *DS*:

Problem DSI: Given finite sets Y_1, Y_2, U_1, U_2, $I \subset Y_1 \times Y_2$ and a function $S: I \rightarrow 2^{U_1 \times U_2}$, are there functions $\gamma_i : Y_i \rightarrow U_i$, $i=1,2$, such that

$$(\gamma_1(y_1), \gamma_2(y_2)) \in S(y_1, y_2), \quad \forall (y_1, y_2) \in I \quad ? \tag{2.2}$$

Note that any instance of *DSI* is equivalent to an instance of *DS* in which

$$S(y_1, y_2) = U_1 \times U_2, \quad \forall (y_1, y_2) \notin I.$$

That is, no compatibility restrictions are placed on the decisions of the two processors for those (y_1, y_2) that cannot occur. We now proceed to the main result of this section:

Proposition 2.2:

a) The problem *DSI* restricted to instances satisfying any of the following:

(i) One or more of $|U_1|$, $|U_2|$, D_1, D_2 is equal to 1,

(ii) $|U_1| = |U_2| = 2$,

(iii) $D_1 = D_2 = 2$,

(iv) $D_1 = |U_2| = 2$, (or $D_2 = |U_1| = 2$),

may be solved in polynomial time.

b) The problem DSI is NP-complete even if we restrict to instances for which $|U_1| = D_1 = 3$, $|U_2| = D_2 = 2$.

Note that the above result draws a precise boundary between polynomial and NP-complete special cases, in terms of D_i, $|U_i|$, $i=1, 2$. We have seen that, in general, *DSI* is NP-complete even if D_1, D_2 are held fixed. There is, however, a special case of *DSI*, with D_1, D_2 constant, for which an efficient algorithm of the dynamic programming type if possible:

Proposition 2.3: Let $Y_1 = Y_2 = \{1,2,...,n\}$. Let D be a positive integer constant. Consider those instances of *DSI* for which $(i,j) \in I$ implies either $|i-j| \leq D$ or $|i-j| \geq n-D$. Then *DSI* may be solved in time which is polynomial in n.

A condition of the type $|i-j| \leq D, \forall (i,j) \in I$ is fairly natural in certain applications. For example, suppose that the observations y_1 and y_2 are noisy measurements of an unknown variable x ($y_i = x + w_i$) where the noises w_i are bounded: $|w_i| < D/2$. Similarly, the condition $|i-j| \leq D$ or $|i-j| \geq n-D$, $\forall (i,j) \in I$, arises if the observations y_1, y_2 are noisy measurements of an unknown quantized angle: $y_i = \theta + w_i \pmod{2\pi}$, where the noises w_i are again bounded $D/2$.

III. DECENTRALIZED DETECTION.

A basic problem in decentralized signal processing, which has attracted much attention recently [1,6,7,13,14,15] is the decentralized detection (hypothesis testing) problem. In this section we consider the discrete version of this problem.

Two processors (sensors) S_1 and S_2 receive measurements $y_i \in Y_i$, $i=1,2$, where Y_i are finite sets. (Figure 3.1). There are two hypotheses H_o and H_1 on the state of the environment with prior probabilities p_o and p_1, respectively. For each hypothesis H_i, we are also given the joint probability mass function $P(y_1,y_2 \mid H_i)$ of the observations, conditioned on the event that H_i is true. Upon receipt of y_i, processor S_i evaluates a binary message $u_i \in \{0,1\}$ according to a rule $u_i = \gamma_i(y_i)$, where γ_i: $Y_i \rightarrow 0,1$. Then, u_1 and u_2 are transmitted to a central processor (fusion center) which evaluates $u_0 = u_1 \wedge u_2$ and declares hypothesis H_o to be true if $u_0=0$, H_1 if $u_0 = 1$. (So, we essentially have a voting scheme). The problem is to select the functions γ_1, γ_2 so as to minimize the probability of accepting the wrong hypothesis. (More general performance criteria may be also considered).

Most available results assume that

$$P(y_1,y_2 \mid H_i) = P(y_1 \mid H_i)\, P(y_2 \mid H_i), \quad i=1,2 \tag{3.1}$$

which states that the observations of the two processors are independent, when conditioned on either hypothesis.* In particular, it has been shown [15] that if (3.1) holds then the optimal decision rules γ_i are given in terms of thresholds for

*Such an assumption is reasonable in problems of detection of a known signal in independent noise, but is typically violated in problems of detection of an unknown signal.

the likelihood ratio $\frac{p_0 P(H_0 \mid y_i)}{p_1 P(H_1 \mid y_i)}$. The optimal thresholds for the two sensors are coupled through a system of equations which gives necessary conditions of optimality. (These equations are just the person-by-person optimality conditions). Few analytical results are available when the conditional inependence assumption is removed [7]. The purpose of this section is precisely to explain this status of affairs.

Suppose that (3.1) holds and let N_i denote the cardinality of Y_i. Given the results of [15] there are only $N_i + 1$ decision rules γ_i which are candidates for being optimal. We may evaluate the cost associated to each pair of candidate decision rules and select a pair with least cost. This corresponds to a polynomial algorithm and shows that under condition (3.1) decentralized detection is an easy problem. Without the conditional independence assumption (3.1), however, there is no guarantee that optimal decision rules can be defined in terms of thresholds for the likelihood ratio. Accordingly, a solution by exhaustive enumeration could require the examination of as many as $2^{N_1+N_2}$ pairs of decision rules. One might expect that a substantially faster (i.e. polynomial) algorithm is possible. However, the main result of this section (Proposition 3.1 below) states that decentralized detection is NP-complete even if we restrict to instances for which perfect detection (zero probability of error) is possible for the corresponding centralized detection problem.

We now present formally a suitable version of the problem:

Decentralized Detection (DD): We are given finite sets Y_1, Y_2; a rational number K; a rational probability mass function $p : Y_1 \times Y_2 \rightarrow Q$; a partition

$\{A_o, A_1\}$ of $Y_1 \times Y_2$. Do there exist $\gamma_i : Y_i \rightarrow \{0,1\}$, $i=1,2$, such that $J(\gamma_1,\gamma_2) \leq K$? Here

$$J(\gamma_1,\gamma_2) = \sum_{(y_1,y_2)\in A_o} p(y_1,y_2)\,\gamma_1(y_1)\,\gamma_2(y_2) + \\ + \sum_{(y_1,y_2)\in A_1} p(y_1,y_2)\,[1-\gamma_1(y_1)\,\gamma_2(y_2)]. \qquad (3.2)$$

Remarks:

1. In the above definition of *DD*, think of H_i as being the hypothesis that $(y_1,y_2) \in A_i$. Then it is easy to see that $J(\gamma_1,\gamma_2)$ corresponds to the probability of error associated to the decision rules γ_1,γ_2. Note that if a single processor knew both y_1 and y_2 (centralized information) he could make the correct decision with certainty. Consequently, the above defined problem corresponds to the special case of decentralized detection problems for which perfect centralized detection is possible.
2. If we let $K = 0$, then *DD* is a special case of problem DS with $|U_1| = |U_2| = 2$ and is therefore polynomially solvable.

Proposition 3.1: *DD* is NP-complete.

It should be pointed out that Proposition 3.1 remains valid if the problem is modified so that fusion center uses some other rule for combining the messages it receives (e.g. $u_o = (u_1 \vee \neg u_2)$), or if the combining rule is unconstrained and the fusion center supposed to find and use an optimal combining rule.

Let us now interpret Proposition 3.1. Although it is, in a sense, a negative result, it can be useful in suggesting meaningful directions for future research:

instead of looking for efficient exact algorithms, the focus should be on approximate ones. (In fact, it is an interesting research question whether polynomial approximate algorithms for *DD* exist.) Proposition 3.2 also shows that any necessary conditions to be developed for problem *DD* will be deficient in one of two respects:

a) Either there will be a very large number of decision rules satisfying these conditions,

b) Or, it will be hard to find decision rules satisfying these conditions.

Another consequence of Proposition 3.1 is that optimal decision rules are not given, in general, in terms of thresholds for the likelihood ratio, because in that case an efficient algorithm could be obtained. Of course, this fact can be also verified directly by constructing appropriate examples. When the condition (3.1) holds and decision rules are given in terms of thresholds, the decision rule of a processor can be viewed as a tentative local decision, submited to the fusion center. In general, however, optimal decision rules are not threshold rules and this interpretation is no more valid. Rather *DD* should be viewed as a problem of optimal quantization of the observation of each processor. In that respect, Proposition 3.1 is reminiscent of the result of [3], namely that the general problem of minimum distortion quantization is NP-comp.ete.

IV. RELATED PROBLEMS.

The best known static decentralized problem is the team decision problem [8,11] which admits an easy and elegant solution under linear quadratic assumptions. Its discrete version is the following:

Team Decision Problem (TDP): Given finite sets Y_1, Y_2, U_1, U_2, a rational probability mass function $p: Y_1 \times Y_2 \rightarrow Q$ and an integer cost function $c: Y_1 \times Y_2 \times U_1 \times U_2 \rightarrow N$, find decision rules $\gamma_i: Y_i \rightarrow U_i$, $i=1,2$, which minimize the expected cost

$$J(\gamma_1,\gamma_2) = \sum_{y_1 \in Y_1} \sum_{y_2 \in Y_2} c(y_1,y_2,\gamma_1(y_1),\gamma_2(y_2))\ p(y_1,y_2)$$

Another problem related to *DS* is the following:

Maximize Probability of Satificing (MPS): Given finite sets Y_1, Y_2, U_1, U_2, a probability mass function $p: Y_1 \times Y_2 \rightarrow Q$ and a function $S: Y_1 \times Y_2 \rightarrow 2^{U_1 \times U_2}$, find decision rules $\gamma_i: Y_i \rightarrow U_i$, $i=1,2$, which maximize

$$J(\gamma_1,\gamma_2) = P((\gamma_1(y_1),\gamma_2(y_2)) \in S(y_1,y_2))$$

(which is the probability of making a satisfactory decision).

Given an instance of *TDP*, let

$$S(y_1,y_2) = \{(u_1,u_2) : c(y_1,y_2,u_1,u_2)=0\}.$$

If we solve *TDP*, we also effectively answer the question whether $J(\gamma_1,\gamma_2)=0$. But this is equivalent to solving the instance of *DS* associated to the above definition of $S(y_1,y_2)$. Therefore, *TDP* cannot be easier than *DS*. The same argument is also valid for *MPS*. It then follows from Proposition 2.2 that *TDP*

and *MPS* are NP-hard (that is, NP-complete or worse) even if we restrict to instances for which $|U_1|=2$, $|U_2|=3$. However, even more is true: it suffices to notice that the problem *DD* of the previous section is a special case of both *TDP* and *MPS*, with $|U_1| = |U_2| = 2$. Using Proposition 3.2, we obtain:

Corollary 4.1: *TDP* and *MPS* are NP-hard even if we restrict to instances for which $|U_1|=|U_2|=2$. This is true even if the cost function c associated to *TDP* is restricted to take only the values 0 and 1.

These results show that unlike the linear quadratic case, the team decision problem is, in general, a hard combinatorial problem.

APPENDIX

Proof of Proposition 2.1: We will reduce to DS (with $|U| = 2, M = 3$) the satisfiability problem of propositional calculus with three literals per clause (3SAT) which is a known NP-complete problem [2]. Given an instance of 3SAT, let V be the set of literals, C the set of clauses. We construct an instance of DS as follows: let

$$Y_i = \{1,2,..., |V|\}, \quad U_i = \{0,1\}, \quad i = 1,2,3.$$

Let

$$S(k,k,k) = \{(0,0,0), (1,1,1)\}, \quad k = 1,...,|V|.$$

Finally, we interpret each clause in C as stating that a certain triple of decisions is not in the satisficing set. (For example, the clause $(x_i \vee \neg x_j \vee x_k)$, translates to the statement that the triple of decisions $(0,1,0)$ does not belong to $S(i,j,k)$.) It is then easy to see that a satisfying assignment for the variables in V exists if and only if the above constructed instance of DS is a YES instance. □

Proof of Proposition 2.2: a)

(i) If $U_1 = 1$ or $U_2 = 1$, the problem is trivial. If $D_2 = 1$, a satisficing decision rule exists if and only if

$$\bigcap_{\{y_2:\,(y_1,y_2)\in I\}} \pi_1(S(y_1,y_2)) \neq \emptyset, \quad \forall y_1 \in Y_1,$$

where $\pi_1(u_1,u_2) \triangleq u_1$. The above condition can be clearly tested in polynomial time (in fact $O(|Y_1|\,|Y_2|)$ time).

(ii) This is the result of [10] mentioned in Section 2.

(iii) Possibly by renaming, let Y_1, Y_2 be disjoint sets. Consider the graph $G = (Y_1 \cup Y_2, I)$. (Here $Y_1 \cup Y_2$ is the set of nodes, I the set of undirected edges.) Since $D_1 = D_2 = 2$, each node of G has degree at most 2. Therefore, the connected components of G are either isolated nodes, chains or cycles. Each connected component of G defines a subproblem and these subproblems are decoupled. So, without loss of generality, we may assume that G consists of a single connected component. We may also assume that G is a cycle (Fig. A.1). (If it is a chain we can introduce an additional edge to make it a cycle; this will not make the problem any easier because a new edge simply allows the presence of some more constraints.) In that case Y_1 and Y_2 have the same number n of elements. Possibly by renumbering (see Fig. A.1) we may assume that

$$I = \{(i,i) : i = 1,\ldots,n\} \cup \{(i,i-1) : i = 2,\ldots,n\} \cup \{(1,n)\}$$

Let us define

$$S'(1,n-1) = \Big\{(u_1,u'_{n-1}) \in U_1 \times U_2 : \exists (u_n,u'_n) \in U_1 \times U_2 \text{ such that}$$

$$(u_n,u'_n) \in S(n,n),\quad (u_n,u'_{n-1}) \in S(n,n-1),\quad (u_1,u'_n) \in S(1,n)$$

and note that $S'(1,n-1)$ can be evaluated in $O(|U_1|^2|U_2|^2)$ time. We now have:

An instance of DSI is a "YES" instance $\Longleftrightarrow$

$\exists (u_1, u'_1, \ldots, u_n, u'_n)$ [$(u_i, u'_i) \in S(i,i)$, $i = 1,\ldots,n$ and

$$(u_i, u'_{i-1}) \in S(i,i-1), \quad i = 2,\ldots,n \quad \text{and}$$

$$(u_1, u'_n) \in S(1,n)\Big] \qquad \Longleftrightarrow$$

$$\exists (u_1, u'_1, \ldots, u_{n-1}, u'_{n-1})\Big[(u_i, u'_i) \in S(i,i), \quad i = 1,\ldots,n-1 \text{ and}$$

$$(u_i, u'_{i-1}) \in S(i,i-1), \quad i = 2,\ldots,n-1 \quad \text{and} \quad (u_1, u'_{n-1}) \in S'(1,n-1)\Big]$$

The last expression corresponds to a new instance of *DSI* in which n has been replaced by $n-1$. Proceeding in the same way, the problem will be solved after at most n similar stages. This is essentially an algorithm of the dynamic programming type, with complexity $O(|Y_1|\ |U_1|^2\ |U_2|^2)$.

Remark: In fact an $O(|Y_1|\ |U_1|\ |U_2|\ (|U_1|+|U_2|))$ solution is possible, if at each stage of the dynamic programming algorithm we only eliminate one - rather than two - variables. Also, if G is a chain, an $O(|Y_1|\ |U_1|\ |U_2|)$ algorithm is obtained along the same lines.

(iv) We now suppose that $D_1 = |U_2| = 2$. Let $Y_1 = \{1,\ldots,m\}$, $Y_2 = \{1,\ldots,n\}$ and assume, without loss of generality, that for each $k \in Y_1$ there exist exactly two distinct elements i, j of Y_2 such that $(k,i) \in I$, $(k,j) \in I$.

Note that we have a "YES" instance of *DSI* if and only if

$$\exists u_1,\ldots,u_n \in U_2\ \exists v_1,\ldots,v_m \in U_1\ \forall (k,i) \in I\ \ [(v_k, u_i) \in S(k,i)]. \tag{A.1}$$

Consider also the statement

$$\exists u_1,\ldots,u_n \in U_2 \; \forall (i,j) \in Y_2 \times Y_2 \; \forall k \in Y_1 \; \Big[[(k,i) \in I \wedge (k,j) \in I \wedge i \neq j] \Longrightarrow$$

$$\exists v_k \in U_1 \; [(v_k,u_i) \in S(k,i) \wedge (v_k,u_j) \in S(k,j)] \Big] \qquad \text{(A.2)}$$

which is equivalent to

$$\exists u_1, \ldots, u_n \in U_2 \; \forall (i,j) \in Y_2 \times Y_2 \; [(u_i,u_j) \in S'(i,j)]. \qquad \text{(A.3)}$$

where

$$S'(i,j) = \Big\{ (u,u') \in U_2 \times U_2 : \forall k \in Y_1 \; \Big[[(k,i) \in I \wedge (k,j) \in I \wedge i \neq j] \Rightarrow$$

$$\exists v_k \in U_1 \; [(v_k,u) \in S(k,i) \wedge (v_k,u') \in S(k,j)] \Big] \Big\} \qquad \text{(A.4)}$$

If (A.1) holds, then it is easy to see that (A.2) holds, as well, with the same assignment of values to u_i, v_k. Conversely, assume that (A.2) holds. For each k, there exists only one pair (i,j) such that the condition $[(k,i) \in I] \wedge [(k,j) \in I] \wedge [(i \neq j)]$ holds. Accordingly, for each k, the clause $[(v_k,u_i) \in S(k,i) \wedge (v_k,u_j) \in S(k,j)]$ needs to be checked only for one pair (i,j). Therefore, for each k, a value of v_k which makes (A.2) true can be chosen regardless of (i,j) and this value makes (A.1) true as well.

Therefore, we only need to show that the truth of (A.2) can be decided in polynomial time. Note that, for each (i,j), the set $S'(i,j)$ defined by (A.4) may be constructed in time $O(|Y_1|\,|U_1|)$. Moreover there are at most $\min\{|Y_2|^2, |Y_1|\}$ pairs to be considered; so the sets $S'(i,j)$ may be constructed in time $O(|Y_1|\,|U_1| \min\{|Y_2|^2, |Y_1|\})$. Once $S'(i,j)$ is constructed, the statement $(u_i,u_j) \in S'(i,j)$ may be expressed as a set of clauses

with two literals per clause (the literals are the boolean variables u_i, u_j; this is similar to the proof of part (ii), see [10]). Therefore, deciding the truth of (A.3) is a special case of the satisfiability problem of propositional calculus with two literals per clause (2SAT), which can be solved in linear time [2].

b) Consider the following restricted version of the satisfiability problem for propositional calculus with 3 literals per clause (3SAT). An instance of this restricted problem (which we call RSAT) consists of the following: a set of boolean variables $F = F_1 \cup F_2$, where $F_1 = \{y_{ij} : i=1,...,m \ ; \ j = 1,2,3\}$, $F_2 = \{x_i : i = 1,...,n\}$; also a set of clauses C consisting of:

(a) one clause for each i between 1 and m, stating that exactly one of the variables y_{i1}, y_{i2}, y_{i3} is true and

(b) an arbitrary number of clauses of the form $(\neg y_{ij} \vee x_k)$ or $(\neg y_{ij} \vee \neg x_k)$. It was shown in [10] that RSAT is NP-complete and that RSAT is equivalent to DS with $|U_1|=3$, $|U_2|=2$. It will be useful to point out the reasons for the latter fact: think of the observation sets Y_1 and Y_2 as being equal to $\{1,...,m\}$ and $\{1,...,n\}$, respectively. Which of the variables y_{i1}, y_{i2}, y_{i3} is true determines whether $\gamma_1(i)$ is equal to 1,2,3; similarly, the value of x_k determines whether $\gamma_2(k)$ is 0 or 1. Finally, a clause of the form $(\neg y_{ij} \vee x_k)$ would indicate that a certain pair of decisions is incompatible, i.e. does not belong to S(i,k).

Let us now introduce a graph with nodes $F_1 \cup F_2$ and edges

$$I = \{(i,k) : \exists\ j \text{ such that } (\neg y_{ij} \vee x_k) \in C \text{ or } (\neg y_{ij} \vee \neg x_k) \in C\}.$$

From this graph we may obtain a new one by lumping together (for each $i \in \{1,...,n\}$) the three nodes y_{i1}, y_{i2}, y_{i3} into one node y_i. Let H_1 (respectively H_2) be the maximum degree of nodes of type y_i (respectively x_k) in this new graph.

We have indicated above a one-to-one correspondence between instances of *DS* and *RSAT*. With this correspondence, it is not hard to see that if we have an instance of DS of degree (D_1,D_2), then the numbers H_1, H_2 obtained from the sociated instance of *RSAT* are actually equal to D_1,D_2, respectively. (The converse is also true.)

For this reason, it only remains to prove the following:

Lemma A.1: *RSAT* is NP-complete, even if it is restricted to instances for which $H_1 = 3$, $H_2 = 2$.

Proof: Given that RSAT is NP-complete [10], it is sufficient to start with a general instance of RSAT and reduce it to a new instance for which $H_1 = 3$, $H_2 = 2$ holds. This is accomplished by creating multiple copies of each variable so that, instead of having the same variable appear in many clauses, distinct copies of it are used. Of course, some clauses will be needed to create the multiple copies but these can be kept to a small number. The idea of the proof is best shown diagramatically, as in Figure A.2. (A more formal argument may be found in [16] but the present one is easier to visualize.)

Let y_{i1}, y_{i2}, y_{i3} be variables in a given instance of RSAT. We introduce some new z-variables, as well as some clauses which guarantee, for example (see

Fig. A.2), that $z_1=y_{i1}$, $z_2=y_{i2}$, $z_3=z_6=z_7=z_{10}=y_{13}$. (For example, the clauses

$$(\neg z_4 \vee \neg z_3) \wedge (\neg z_5 \vee \neg z_3) \wedge (\neg z_6 \vee z_3),$$

together with a requirement that exactly one of the variables z_4, z_5, z_6 is true, implies $z_3=z_6$.)* In this way we can effectively create an arbitrary number of copies of the y-variables and the same procedure works for the x-variables as well. Note that in doing so, we have respected the requirement $H_1 = 3$, $H_2 = 2$. Finally, for each clause in the original instance of RSAT, we may introduce a clause involving appropriate copies and it is easy to see that the requirement $H_1 = 3$, $H_2 = 2$ will be still respected, as long as we use a different copy each time. Moreover, since an arbitrary number of copies may be created by the above procedure, an arbitrary number of clauses for the original instance of RSAT may be thus handled. □

Proof of Proposition 2.3: This proof is effectively a generalization of the dynamic programming argument in the proof of part (a) (iii) of Proposition 2.2. Let us assume that $n \geq 2D$. For any k such that $2D \leq k \leq n$, let

$$\Gamma(k) = \Big\{(u_1,v_1,u_2,v_2,\ldots,u_D,v_D,u_{k-D+1},v_{k-D+1},\ldots,u_k,v_k) \in (U_1 \times U_2)^{2D} :$$

$$\exists\ (u_{D+1},v_{D+1},\ldots,u_{k-D},v_{k-D}) \in (U_1 \times U_2)^{k-2D} \text{ such that}$$

* The arcs in Figure A.2 indicate the variables that have a common clause; solid lines indicate that two variables are constrained to be equal; a curve encircling a triple of variables in F_1 indicates that these are to be merged to a single node.

$$(u_i, v_j) \in S(i,j), \forall (i,j) \in \{1,\ldots,k\}^2 \cap I \quad \Big\}$$

Note that $\Gamma(k)$ is of size at most $|U_1|^{2D} |U_2|^{2D}$. Now assume that $2D \leq k \leq n-1$ and let

$$\hat{\Gamma}(k+1) = \Big\{ (u_1,v_1,u_2,v_2,\ldots,u_D,v_D,u_{k-D+1},v_{k-D+1},\ldots,u_{k+1},v_{k+1}) \in (U_1 \times U_2)^{2D+2} :$$
$$\exists\ (u_{D+1},v_{D+1},\ldots,u_{k-D},v_{k-D}) \in (U_1 \times U_2)^{k-2D} \text{ such that}$$
$$(u_i,v_j) \in S(i,j), \quad \forall (i,j) \in \{1,\ldots,k+1\}^2 \cap I \quad \Big\}$$

Using the assumption $|i-j| \leq D$, or $|i-j| \geq n-D$, $\forall (i,j) \in I$, we can see that

$$\{1,\ldots,k+1\}^2 \cap I \subset [\{1,\ldots,k\}^2 \cap I] \cup A_{k+1}$$

where

$$A_{k+1} = \{1,\ldots,D,k-D+1,\ldots,k+1\}^2 .$$

With this observation, $\hat{\Gamma}(k+1)$ may be rewritten as

$$\hat{\Gamma}(k+1) =$$
$$\Big\{ (u_1,v_1,u_2,v_2,\ldots,u_D,v_D,u_{k-D+1},v_{k-D+1},\ldots,u_{k+1},v_{k+1}) \in (U_1 \times U_2)^{2D+2} :$$
$$(u_1,v_1,u_2,v_2,\ldots,u_D,v_D,u_{k-D+1},v_{k-D+1},\ldots,u_k,v_K) \in \Gamma(k) \text{ and}$$
$$(u_i,v_j) \in S(i,j), \forall (i,j) \in A_{k+1} \Big\}$$

Assuming that the set $\Gamma(k)$ has been computed, we may use it to evaluate $\hat{\Gamma}(k+1)$ as follows: for each element of $\Gamma(k)$ (at most $|U_1|^{2D} |U_2|^{2D}$ elements) try each pair $(u_{k+1},v_{k+1}) \in U_1 \times U_2$ ($|U_1| |U_2|$ pairs) and check for each $(i,j) \in A_{k+1} \cap I$ (A_{k+1} has at most $4D^2$ elements) whether $(u_i,v_j) \in$

$S(i,j)$ holds. Therefore, given $\Gamma(k)$, we may obtain $\hat{\Gamma}(k+1)$ in time $O\,(D^2 \mid U_1 \mid^{2D+1} \mid U_2 \mid^{2D+1})$. Finally, from $\hat{\Gamma}(k+1)$ we may easily obtain $\Gamma(k+1)$, by taking a projection so as to eliminate u_{k-D+1}, v_{k-D+1} . This process may be repeated (for no more than n stages) to compute $\Gamma(n)$, in time $O(nD^2 \mid U_1 \mid^{2D+1} \mid U_2 \mid^{2D+1})$. Then note that we have a YES instance of *DSI* if and only if $\Gamma(n) \neq \emptyset$. □

Remark: The algorithm in the proof of Proposition 2.3 does not find a satisficing decision rule; it only determines whether one exists. However, satisificing decisions rules may be computed by keeping in the memory some of the intermediate results produced by the algorithm.

Proof of Proposition 3.1: Consider the following problem of propositional calculus, which we call P:

Problem P: We are given two sets $X = \{x_1,...,x_m\}$, $Z=\{z_1,...,z_n\}$ of boolean variables; a set D of (distinct) clauses of the form $x_i \wedge z_j$ or $\neg(x_i \wedge z_j)$ (we assume that for any pair (i,j) at most one of the above clauses is in D); a collection $\{q_{ij} : i \in \{1,...,m\},\ j \in \{1,...,n\}\}$ of non-negative integers and an integer K. Is there a truth assignment for X and Z such that $J \leq K$, where

$$J \triangleq \sum_{\substack{x_i \wedge z_j = 0 \\ (i,j) \in A_1}} q_{ij} + \sum_{\substack{x_i \wedge z_j = 1 \\ (i,j) \in A_0}} q_{ij}\ , \tag{A.5}$$

$$A_0 = \{(i,j)\text{: the clause } \neg(x_i \wedge z_j) \text{ is in } D\},$$

$$A_1 = \{(i,j)\text{: the clause } (x_i \wedge z_j) \text{ is in } D\}^*$$

* So, J is the sum of the weights q_{ij} of the clauses that are not satisfied.

Lemma A.2: Problem P is equivalent to DD.

Proof: Think of X, Z as being the sets of observations of processors S_1, S_2, respectively. A truth assignment to X,Z corresponds to a choice as to what binary message to transmit to the fusion center, given each processor's observation. Let H_o (respectively H_1) be the hypothesis that $(i,j) \in A_o$ (respectively A_1). Finally, view q_{ij}, as the (unnormalized) probability that the pair (i,j) of observations is obtained by the two processors. Pairs (i,j) that belong to neither A_o nor A_1 may be viewed as having zero probability and are, therefore, of no concern. Then, it is easy to verify that J, as defined by (A.5) is precisely the (unnormalized) probability of error. □

In order to complete the proof of the Proposition, we need to show that P is NP-complete. This will be accomplished by reducing to P the following (Maximum 2-Satisfiability) problem of propositional calculus which is known to be NP-complete [2]:

MAX-2-SAT: Given a set U of boolean variables, a collection C of (distinct) clauses over U, such that each clause $c \in C$ has exactly two variables and an integer $K \leq | C |$, is there a truth assignment for U which simultaneously satisfies at least K of the clauses in C? (Without loss of generality, we assume that if a clause is in C, then its negation is not in C).

Suppose that we are given an instance (U,C,K) of MAX-2-SAT. We construct an instance of P as follows: Suppose that $U = \{u_1,...,u_n\}$. Then, let $X=\{x_{i1},x_{i2},x_{i3}: i=1,...,n\}$ and $Z = \{z_{i1},z_{i2},z_{i3} : i = 1,...,n\}$. For each

$i \in \{1,...,n\}$ introduce the set D_i of clauses:

$$\neg(x_{i1} \wedge z_{i2}),\quad (x_{i2} \wedge z_{i2}),\quad \neg(x_{i2} \wedge z_{i1}),\quad (x_{i3} \wedge z_{i2}),$$

$$\neg(x_{i2} \wedge z_{i3}),\quad (x_{i3} \wedge z_{i1}),\quad (x_{i1} \wedge z_{i3}),\quad (x_{i3} \wedge z_{i3}) \quad .$$

To these clauses we assign the weights (L is a large integer to be determined later):

$$q_{i1,i2} = 30L \quad q_{i2,i2} = 15L, \quad q_{i2,i1} = 4L, \quad q_{i3,i2} = 20L$$

$$q_{i2,i3} = 8L, \quad q_{i3,i1} = 2L, \quad q_{i1,i3} = 25L, \quad q_{i3,i3} = 100L \quad .$$

Next, for each clause $(u_i \wedge u_j)$, $(\neg u_i \wedge u_j)$, $(\neg u_i \wedge \neg u_j)$, $(u_i \vee u_j)$, $(\neg u_i \vee u_j)$, $(\neg u_i \vee \neg u_j)$ in C (with $i < j$), introduce clauses $(x_{i1} \wedge z_{j1})$, $(x_{i2} \wedge z_{j1})$, $(x_{i2} \wedge z_{j2})$, $\neg(x_{i2} \wedge z_{j2})$, $\neg(x_{i1} \wedge z_{j2})$, $\neg(x_{i1} \wedge z_{j1})$, respectively. Denote this last set of clauses by D_o, and assign to each one unit weight. We now let $D = \bigcup_{l=0}^{n} D_l$ and observe that $X,Z,D,\{q_{ij}\}$, K define an instance of P.

Note that for any assignment for X,Z, the corresponding cost (equation (A.5)) may be decomposed as

$$J = J_o + \sum_{l=1}^{n} J_l \ , \tag{A.6}$$

where J_l, $l \in \{0,1,...,n\}$ is the sum of the weights q_{ij} of the clauses in D_l which are not satisfied.

Lemma A.3: For any $i \in \{1,...,n\}$, we have $J_i = 35L$ if and only if either of

the following is true:

$$(i) \quad x_{i1} = z_{i1} = x_{i3} = z_{i3} = 1 \ , \quad x_{i2} = z_{i2} = 0 \quad .$$

$$(ii) \quad x_{i2} = z_{i2} = x_{i3} = z_{i3} = 1, \quad x_{i1} = z_{i1} = 0 \quad .$$

For any assignment to $\{x_{ij}, z_{ij} : i = 1,2,3\}$ other than the two assignments above, $J_i \geq 37L$.

Proof: By direct evaluation of J_i for each possible assignment. See [16]. □

In view of Lemma A.3, the clauses in D_i and their associated weights have the following interpretation: the variable x_{i1} may be freely assigned, but the remaining variables must be assigned so that $x_{i2} = z_{i2} = \neg z_{i1} = \neg x_{i1}$. For this reason the clauses in D_o are effectively the same as the original set C of clauses.

Lemma A.4: Let L be large enough so that $|C| < L$. Then, there exits a truth assignment for U for which at least K clauses in C are satisfied, if and only if there exists a truth assignment for X,Z such that the resulting cost J is less or equal than $35nL + |C| - K$.

Proof: (i) Given an assignment for U, with at least K clauses satisfied, assign the variables in X,Z as follows:

$$x_{i1} = z_1 = u_1, \quad x_{i2} = z_2 = \neg u_1, \quad x_{i3} = z_{i3} = 1.$$

Using Lemma A.3 and the identity (A.6), the resulting cost is $35nL$ (i.e. $35L$ from each collection D_i, $i=1,...,n$) plus the number of clauses in D_o which are not satisfied (since these carry unit weight). The latter number is identical to the number of clauses in C which are not satisfied, which is less or equal than $|C| - K$.

(ii) Conversely, given an assignment for X,Z such that $J \leq 35nL + \mid C \mid -K$, suppose that for some $i \in \{1,...,n\}$, $J_i \geq 37L$. Using Lemma A.3 and the inequality $\mid C \mid < L$, we obtain

$$J \geq \sum_{i=1}^{n} J_i \geq 35nL + 2L > 35nL + \mid C \mid -K$$

which is a contradiction and shows that $J_i = 35L$, $\forall\, i$. Consequently, $\{x_{i1},x_{i2},z_{i1},z_{i2}\}$ have been assigned values in one of the two ways suggested by Lemma A.3. We now assign truth values for U, by setting $u_i = x_{i1}$. Then J_o is the number of clauses in C which are not satisfied. Moreover, since $J_i = 35L, i \in \{1,...,n\}$, it follows that $J_o \leq \mid C \mid -K$, which implies that at least K clauses in C are satisfied. This completes the proof of Lemma A.4. □

It is easy to see that the above reduction of *MAX-2-SAT* to P is polynomial. Therefore, P is NP-complete and so is DD, thus completing the proof of the proposition. □

REFERENCES

1. Ekchian, L.K., "Distributed Detection and Communication Problems", Ph.D. Thesis, Dept. of Electrical Engineering and Computer Science, MIT, Cambridge, MA, 1982.
2. Garey, M.R. and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Co., San Francisco, CA, 1979
3. Garey, M.R., D.S. Johnson and H.S. Witsenhausen, "The Complexity of the Generalized Lloyd-Max Problem", *IEEE Transactions on Information Theory*, Vol. IT-28, No. 2, 1982, pp. 255-256.
4. Ho, Y.C., "Team Decision Theory and Information Structure", *IEEE Proceedings*, Vol. 68, No. 6, 1980, pp. 644-654.
5. Ho, Y.C. and T.S. Chang, "Another Look at the Nonclassical Information Problem", *IEEE Transactions on Automatic Control*, Vol. AC-25, No. 3, 1980, pp. 537-540.
6. Kushner, H.J., and A. Pacut, "A Simulation Study of a Decentralized Detection Problem", *IEEE Transactions on Automatic Control*, Vol. AC-27, No. 5, 1982, pp. 1116-1119.
7. Lauer, G.S. and N.R. Sandell, Jr., "Distributed Detection of Signal Waveforms in Additive Gaussian Observation Noise", Technical Paper 160, Alphatech Inc., Burlington, MA, 1983.
8. Marschak, J.R. and R. Radner, *The Economic Theory of Teams*, Yale University Press, New Haven, CT, 1972.
9. Papadimitriou, C.H. and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall, Englewood Cliffs, NJ, 1982.
10. Papadimitriou, C.H. and J.N. Tsitsiklis, "On the Complexity of Designing Distributed Protocols", *Information and Control*, Vol. 53, No. 3, June 1982, pp. 211-218.
11. Radner, R., "Team Decision Problems", *Annals of Mathematical Statistics*, Vol. 33, 1962, pp. 857-881.
12. Sandell, N.R., Jr., P. Varaiya, M. Athans and M. Safonov, "Survey of Decentralized Control Methods for Large Scale Systems", *IEEE Transactions on Automatic Control*, Vol. AC-23, No. 2, 1978, pp. 108-128.
13. Teneketzis, D., "The Decentralized Quickest Detection Problem", *Proceedings of the 22nd IEEE Conference on Decision and Control*, Orlando, FL, 1982.
14. Teneketzis, D. "The Decentralized Wald Problem", *Proceedings of the 1983 American Control Conference*, San Francisco, CA, 1983.
15. Tenney, R.R. and N.R. Sandell, Jr., "Detection with Distributed Sensors", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-17, No. 4, 1981, pp. 501-509.
16. Tsitsiklis, J.N., "Problems in Decentralized Decision Making and Computation", Ph.D. Thesis, Dept. of Electrical Engineering and Computer Science, MIT, Cambridge, MA, in preparation.
17. Witsenhausen, H.S., "A Counterexample in Stochastic Optimum Control", *SIAM J. Control* Vol. 6, No. 1, 1968, pp. 138-147.

Figure 3.1

A Structure for Decentralized Detection

Figure A.1

A Bipartite Graph Consisting of a Single Cycle

Figure A.2

The Construction of Copies of the Original Variables

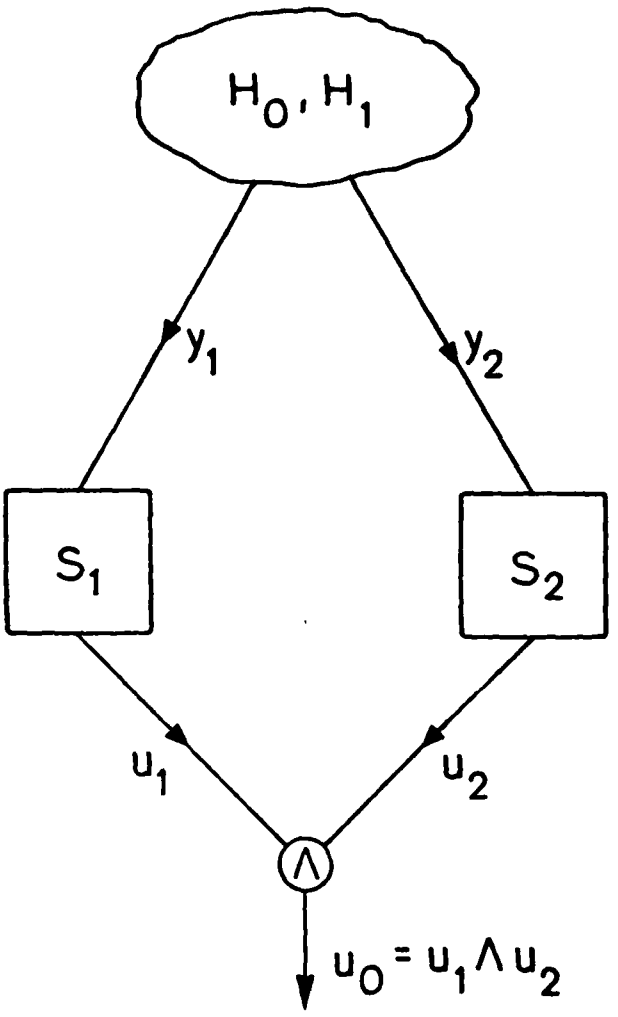


Figure 3.1

A Structure for Decentralized Detection

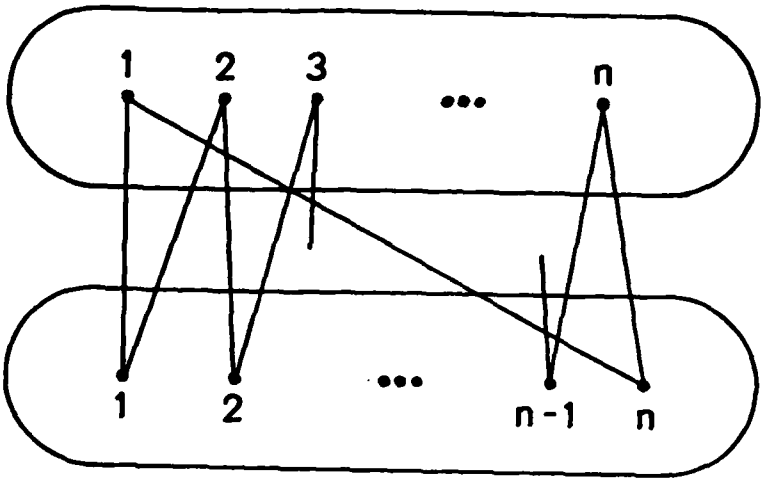


Figure A.1

A Bipartite Graph Consisting of a Single Cycle

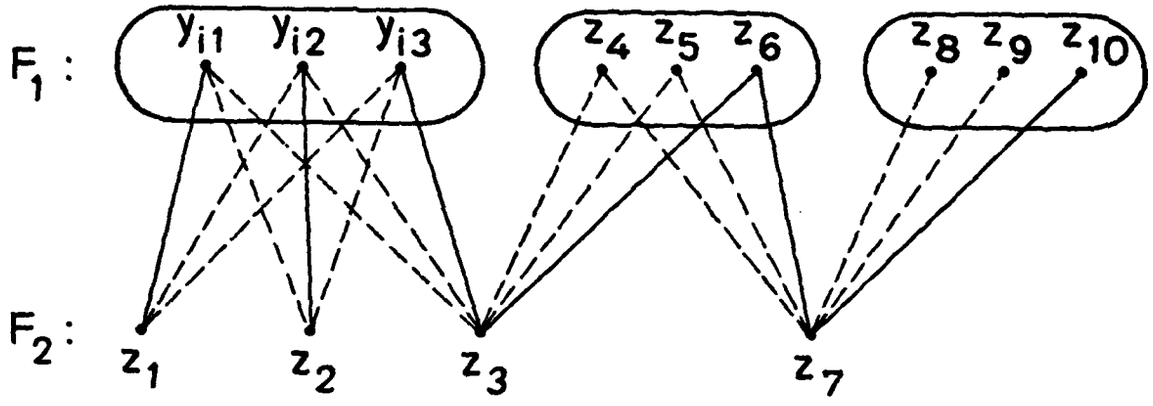
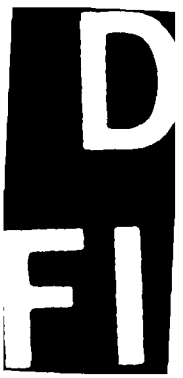


Figure A.2

The Construction of Copies of the Original Variables


D
FI